Conteúdo

Prefácio **xi**

0.1 Exercícios sobre classes xiii

1 Introdução à Programação **1**

1.1 Computadores 1

1.2 Programação 2

1.3 Algoritmos: resolvendo problemas 3

1.3.1 Regras do jogo 4

1.3.2 Desenvolvimento e demonstração de correcção 10

1.4 Programas 16

1.5 Resumo: resolução de problemas 17

2 Conceitos básicos de programação **19**

2.1 Introdução 19

2.1.1 Consola e canais 22

2.1.2 Definição de variáveis 25

2.1.3 Controlo de fluxo 26

2.2 Variáveis 27

2.2.1 Memória e inicialização 27

2.2.2 Nomes de variáveis 29

2.2.3 Inicialização de variáveis 29

2.3 Tipos básicos 30

2.3.1 Tipos aritméticos 32

2.3.2 Booleanos ou lógicos 38

2.3.3 Caracteres 38

2.4 Valores literais . . . 40
2.5 Constantes . . . 41
2.6 Instâncias . . . 43
2.7 Expressões e operadores . . . 43
2.7.1 Operadores aritméticos . . . 44
2.7.2 Operadores relacionais e de igualdade . . . 46
2.7.3 Operadores lógicos . . . 47
2.7.4 Operadores *bit*-a-*bit* . . . 48
2.7.5 Operadores de atribuição . . . 50
2.7.6 Operadores de incrementação e decrementação . . . 51
2.7.7 Precedência e associatividade . . . 52
2.7.8 Efeitos laterais e mau comportamento . . . 52
2.7.9 Ordem de cálculo . . . 54

3 Modularização: funções e procedimentos 57
3.1 Introdução à modularização . . . 57
3.2 Funções e procedimentos: rotinas . . . 60
3.2.1 Abordagens descendente e ascendente . . . 61
3.2.2 Definição de rotinas . . . 68
3.2.3 Sintaxe das definições de funções . . . 71
3.2.4 Contrato e documentação de uma rotina . . . 72
3.2.5 Integração da função no programa . . . 74
3.2.6 Sintaxe e semântica da invocação ou chamada . . . 75
3.2.7 Parâmetros . . . 76
3.2.8 Argumentos . . . 77
3.2.9 Retorno e devolução . . . 77
3.2.10 Significado de `void` . . . 77
3.2.11 Passagem de argumentos por valor e por referência . . . 79
3.2.12 Variáveis locais e globais . . . 87
3.2.13 Blocos de instruções ou instruções compostas . . . 87
3.2.14 Âmbito ou visibilidade de variáveis . . . 88
3.2.15 Duração ou permanência de variáveis . . . 91
3.2.16 Nomes de rotinas . . . 92

3.2.17 Declaração vs. definição . . . 93
3.2.18 Parâmetros constantes . . . 97
3.2.19 Instruções de asserção . . . 100
3.2.20 Melhorando módulos já produzidos . . . 110
3.3 Rotinas recursivas . . . 113
3.4 Mecanismo de invocação de rotinas . . . 115
3.5 Sobrecarga de nomes . . . 122
3.6 Parâmetros com argumentos por omissão . . . 124

4 Controlo do fluxo dos programas **127**
4.1 Instruções de selecção . . . 127
4.1.1 As instruções `if` e `if else` . . . 128
4.1.2 Instruções de selecção encadeadas . . . 131
4.1.3 Problemas comuns . . . 134
4.2 Asserções . . . 135
4.2.1 Dedução de asserções . . . 135
4.2.2 Predicados mais fortes e mais fracos . . . 137
4.2.3 Dedução da pré-condição mais fraca de uma atribuição . . . 137
4.2.4 Asserções em instruções de selecção . . . 139
4.3 Desenvolvimento de instruções de selecção . . . 144
4.3.1 Escolha das instruções alternativas . . . 145
4.3.2 Determinação das pré-condições mais fracas . . . 145
4.3.3 Determinação das guardas . . . 146
4.3.4 Verificação das guardas . . . 147
4.3.5 Escolha das condições . . . 147
4.3.6 Alterando a solução . . . 148
4.3.7 Metodologia . . . 149
4.3.8 Discussão . . . 150
4.3.9 Outro exemplo de desenvolvimento . . . 153
4.4 Variantes das instruções de selecção . . . 156
4.4.1 O operador `? :` . . . 156
4.4.2 A instrução `switch` . . . 157
4.5 Instruções de iteração . . . 162

4.5.1 A instrução de iteração `while` . . . 162
4.5.2 Variantes do ciclo `while`: `for` e `do while` . . . 165
4.5.3 Exemplo simples . . . 171
4.5.4 `return`, `break`, `continue`, e `goto` em ciclos . . . 173
4.5.5 Problemas comuns . . . 181
4.6 Asserções com quantificadores . . . 181
4.6.1 Somas . . . 182
4.6.2 Produtos . . . 183
4.6.3 Conjunções e o quantificador universal . . . 184
4.6.4 Disjunções e o quantificador existencial . . . 185
4.6.5 Contagens . . . 186
4.6.6 O resto da divisão . . . 187
4.7 Desenvolvimento de ciclos . . . 187
4.7.1 Noção de invariante . . . 191
4.7.2 Correcção de ciclos . . . 198
4.7.3 Melhorando a função `potência()` . . . 201
4.7.4 Metodologia de Dijkstra . . . 202
4.7.5 Um exemplo . . . 215
4.7.6 Outro exemplo . . . 227

5 Matrizes, vectores e outros agregados 233
5.1 Matrizes clássicas do C++ . . . 234
5.1.1 Definição de matrizes . . . 235
5.1.2 Indexação de matrizes . . . 236
5.1.3 Inicialização de matrizes . . . 238
5.1.4 Matrizes multidimensionais . . . 239
5.1.5 Matrizes constantes . . . 240
5.1.6 Matrizes como parâmetros de rotinas . . . 240
5.1.7 Restrições na utilização de matrizes . . . 245
5.2 Vectores . . . 246
5.2.1 Definição de vectores . . . 247
5.2.2 Indexação de vectores . . . 247
5.2.3 Inicialização de vectores . . . 248

5.2.4 Operações . . . 249
5.2.5 Acesso aos itens de vectores . . . 249
5.2.6 Alteração da dimensão de um vector . . . 250
5.2.7 Inserção e remoção de itens . . . 251
5.2.8 Vectores multidimensionais? . . . 252
5.2.9 Vectores constantes . . . 253
5.2.10 Vectores como parâmetros de rotinas . . . 253
5.2.11 Passagem de argumentos por referência constante . . . 255
5.2.12 Outras operações com vectores . . . 259
5.3 Algoritmos com matrizes e vectores . . . 260
5.3.1 Soma dos elementos de uma matriz . . . 260
5.3.2 Soma dos itens de um vector . . . 264
5.3.3 Índice do maior elemento de uma matriz . . . 264
5.3.4 Índice do maior item de um vector . . . 270
5.3.5 Elementos de uma matriz num intervalo . . . 270
5.3.6 Itens de um vector num intervalo . . . 275
5.3.7 Segundo elemento de uma matriz com um dado valor . . . 276
5.3.8 Segundo item de um vector com um dado valor . . . 284
5.4 Cadeias de caracteres . . . 286
5.4.1 Cadeias de caracteres clássicas . . . 287
5.4.2 A classe `string` . . . 289

6 Tipos enumerados **295**
6.1 Sobrecarga de operadores . . . 297

7 Tipos abstractos de dados e classes C++ **299**
7.1 De novo a soma de fracções . . . 300
7.2 Tipos Abstractos de Dados e classes C++ . . . 305
7.2.1 Definição de TAD . . . 306
7.2.2 Acesso aos membros . . . 309
7.2.3 Alguma nomenclatura . . . 309
7.2.4 Operações suportadas pelas classes C++ . . . 310
7.3 Representação de racionais por fracções . . . 310

7.3.1 Operações aritméticas elementares . . . 311
7.3.2 Canonicidade do resultado . . . 311
7.3.3 Aplicação à soma de fracções . . . 312
7.3.4 Encapsulamento e categorias de acesso . . . 316
7.3.5 Rotinas membro: operações e métodos . . . 317
7.4 Classes C++ como módulos . . . 323
7.4.1 Construtores . . . 324
7.4.2 Construtores por cópia . . . 330
7.4.3 Condição invariante de classe . . . 331
7.4.4 Porquê o formato canónico das fracções? . . . 332
7.4.5 Explicitação da condição invariante de classe . . . 334
7.5 Sobrecarga de operadores . . . 342
7.6 Testes de unidade . . . 345
7.7 Devolução por referência . . . 349
7.7.1 Mais sobre referências . . . 349
7.7.2 Operadores ++ e −− prefixo . . . 357
7.7.3 Operadores ++ e −− sufixo . . . 359
7.8 Mais operadores para o TAD `Racional` . . . 362
7.8.1 Operadores de atribuição especiais . . . 362
7.8.2 Operadores aritméticos . . . 367
7.9 Construtores: conversões implícitas e valores literais . . . 369
7.9.1 Valores literais . . . 369
7.9.2 Conversões implícitas . . . 369
7.9.3 Sobrecarga de operadores: operações ou rotinas? . . . 370
7.10 Operadores igualdade, diferença e relacionais . . . 371
7.10.1 Inspectores e interrogações . . . 372
7.10.2 Operadores de igualdade e diferença . . . 373
7.10.3 Operadores relacionais . . . 374
7.11 Constância: verificando erros durante a compilação . . . 375
7.11.1 Passagem de argumentos . . . 376
7.11.2 Constantes implícitas: operações constantes . . . 381
7.11.3 Devolução por valor constante . . . 385

7.11.4 Devolução por referência constante 388
7.12 Reduzindo o número de invocações com `inline` 389
7.13 Optimização dos cálculos com racionais 394
7.13.1 Adição e subtracção 395
7.13.2 Multiplicação 398
7.13.3 Divisão 399
7.13.4 Simétrico e identidade 400
7.13.5 Operações de igualdade e relacionais 400
7.13.6 Operadores especiais 400
7.14 Operadores de inserção e extracção 402
7.14.1 Sobrecarga do operador << 403
7.14.2 Sobrecarga do operador >> 405
7.14.3 Lidando com erros 407
7.14.4 Coerência entre os operadores << e >> 409
7.14.5 Leitura e escrita de ficheiros 411
7.15 Amizades e promiscuidades 415
7.15.1 Rotinas amigas 415
7.15.2 Classes amigas 417
7.15.3 Promiscuidades 418
7.16 Código completo do TAD `Racional` 418
7.17 Outros assuntos acerca de classes C++ 432
7.17.1 Constantes membro 432
7.17.2 Membros de classe 433
7.17.3 Destrutores 436
7.17.4 De novo os membros de classe 437
7.17.5 Construtores por omissão 438
7.17.6 Matrizes de classe 441
7.17.7 Conversões para outros tipos 442
7.17.8 Uma aplicação mais útil das conversões 444

8 Programação baseada em objectos **447**
8.1 Desenho de classes 449

9 Modularização de alto nível **451**

9.1 Modularização física 453

9.1.1 Constituição de um módulo 454

9.2 Fases da construção do ficheiro executável 454

9.2.1 Pré-processamento 455

9.2.2 Compilação 462

9.2.3 Fusão 464

9.2.4 Arquivos 467

9.3 Ligação dos nomes 470

9.3.1 Vantagens de restringir a ligação dos nomes 471

9.3.2 Espaços nominativos sem nome 474

9.3.3 Ligação de rotinas, variáveis e constantes 476

9.3.4 Ligação de classes e tipos enumerados 483

9.4 Conteúdo dos ficheiros de interface e implementação 487

9.4.1 Relação entre interface e implementação 488

9.4.2 Ferramentas de utilidade interna ao módulo 488

9.4.3 Rotinas não-membro 489

9.4.4 Variáveis globais 489

9.4.5 Constantes globais 489

9.4.6 Tipos enumerados não-membro 490

9.4.7 Classes não-membro 491

9.4.8 Métodos (rotinas membro) 491

9.4.9 Variáveis e constantes membro de instância 492

9.4.10 Variáveis membro de classe 492

9.4.11 Constantes membro de classe 493

9.4.12 Classes membro (embutidas) 493

9.4.13 Enumerados membro 494

9.4.14 Evitando erros devido a inclusões múltiplas 495

9.4.15 Ficheiro auxiliar de implementação 497

9.5 Construção automática do ficheiro executável 497

9.6 Modularização em pacotes 504

9.6.1 Colisão de nomes 504

9.6.2 Espaços nominativos . . . 506
9.6.3 Directivas de utilização . . . 507
9.6.4 Declarações de utilização . . . 509
9.6.5 Espaços nominativos e modularização física . . . 510
9.6.6 Ficheiros de interface . . . 511
9.6.7 Ficheiros de implementação e ficheiros auxiliares de implementação . . . 512
9.6.8 Pacotes e espaços nominativos . . . 512
9.7 Exemplo final . . . 513

10 Listas e iteradores **519**
10.1 Listas . . . 519
10.1.1 Operações com listas . . . 520
10.2 Iteradores . . . 523
10.2.1 Operações com iteradores . . . 524
10.2.2 Operações se podem realizar com iteradores e listas . . . 525
10.2.3 Itens fictícios . . . 525
10.2.4 Operações que invalidam os iteradores . . . 529
10.2.5 Conclusão . . . 529
10.3 Interface . . . 530
10.3.1 Interface de `ListaInt` . . . 530
10.3.2 Interface de `ListaInt::Iterador` . . . 535
10.3.3 Usando a interface das novas classes . . . 538
10.3.4 Teste dos módulos . . . 540
10.4 Implementação simplista . . . 541
10.4.1 Implementação de `ListaInt` . . . 542
10.4.2 Implementação de `ListaInt::Iterador` . . . 544
10.4.3 Implementação dos métodos públicos de `ListaInt` . . . 547
10.4.4 Implementação dos métodos públicos de `ListaInt::Iterador` . . . 551
10.5 Uma implementação mais eficiente . . . 552
10.5.1 Cadeias ligadas . . . 553

11 Ponteiros e variáveis dinâmicas **557**

12 Herança e polimorfismo **559**

13 Programação genérica **561**

14 Excepções e tratamento de erros **563**

A Notação e símbolos **581**

A.1 Notação e símbolos 581

A.2 Abreviaturas e acrónimos 584

B Um pouco de lógica **585**

C Curiosidades e fenómenos estranhos **587**

C.1 Inicialização 587

C.1.1 Inicialização de membros 588

C.2 Rotinas locais 588

C.3 Membros acessíveis "só para leitura" 589

C.4 Variáveis virtuais 591

C.5 Persistência simplificada 597

D Nomes e seu formato: recomendações **609**

E Palavras-chave do C++ **611**

F Precedência e associatividade no C++ **615**

G Tabelas de codificação ISO-8859-1 (Latin-1) e ISO-8859-15 (Latin-9) **619**

H Listas e iteradores: listagens **625**

H.1 Versão simplista 625

H.1.1 Ficheiro de interface: `lista_int.H` 625

H.1.2 Ficheiro de implementação auxiliar: `lista_int_impl.H` 630

H.1.3 Ficheiro de implementação: `lista_int.C` 635

I Listas e iteradores seguros **643**